INTEGRATING DIMENSION REDUCTION AND AUGMENTATION METHODS FOR ENHANCED FINGERPRINT CLASSIFICATION

PROJECT WORK II PHASE II REPORT

Submitted by

HARISH C
21ADR016

MOHAMED RIZWAN M
21ADR028

MONISHA K
21ADR029

in partial fulfilment of the requirements

for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

ARTIFICIAL INTELLIGENCE
AND DATA SCIENCE

DEPARTMENT OF ARTIFICIAL INTELLIGENCE

KONGU ENGINEERING COLLEGE
(Autonomous)

PERUNDURAI, ERODE - 638060

MARCH 2025

DEPARTMENT OF ARTIFICIAL INTELLIGENCE

KONGU ENGINEERING COLLEGE
(Autonomous)

PERUNDURAI, ERODE – 638060

MARCH 2025

BONAFIDE CERTIFICATE

This is to certify that the Project Report titled **INTEGRATING DIMENSION REDUCTION AND AUGMENTATION METHODS FOR ENHANCED FINGERPRINT CLASSIFICATION** is the bonafide record of project work done by **HARISH C(21ADR016), MOHAMED RIZWAN M(21ADR028), MONISHA K (21ADR029)** in partial fulfilment of the requirements for the award of Degree of Bachelor of Technology in Artificial Intelligence of Anna University, Chennai during the year 2024-2025.

SUPERVISOR **HEAD OF THE DEPARTMENT**

(Signature with seal)

Date:

Submitted for the end semester viva voce examination held on______________.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DEPARTMENT OF ARTIFICIAL INTELLIGENCE

KONGU ENGINEERING COLLEGE
(Autonomous)

PERUNDURAI, ERODE – 638 060

MARCH 2025

DECLARATION

We affirm that the Project Report titled **INTEGRATING DIMENSION REDUCTION AND AUGMENTATION METHODS FOR ENHANCED FINGERPRINT CLASSIFICATION** being submitted in partial fulfilment of the requirements for the award of Bachelor of Technology is the original work carried out by us. It has not formed part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Date:

HARISH C
(21ADR016)

MOHAMED RIZWAN M
(21ADR028)

MONISHA K
(21ADR029)

I certify that the declaration made by the above candidate is true to the best of my knowledge.

Date: Name & Signature of the Supervisor with seal

ABSTRACT

Fingerprint classification remains a crucial aspect of biometric authentication, but conventional limitations in data availability restrict model performance. This research employs state-of-the-art deep learning techniques such as inversion and multi-augmentation to generate a rich set of training samples. To optimize feature selection, dimensionality reduction methods such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) maximize the combination of feature maps, improving classification accuracy.

The proposed work explores pretrained Convolutional Neural Networks (CNNs) such as AlexNet and GoogleNet, along with more complex architectures such as Recurrent Neural Networks (RNNs) and Deep Belief Networks (DBNs), enhancing feature extraction and model generalization. By combining augmentation and dimensionality reduction techniques, this research develops an efficient, high-performance system for fingerprint identification. The proposed system optimizes classification efficacy while maintaining computational efficiency, ensuring suitability for real-world biometric applications.

Additionally, the research examines trade-offs among various deep learning architectures, focusing on the impact of multi-augmentation and feature selection on fingerprint classification. Experimental results demonstrate that the optimized combination of augmentation, dimensionality reduction, and classifier selection maximally enhances model robustness and generalization, advancing biometric authentication systems.

ACKNOWLEDGEMENT

First and foremost, we acknowledge the abundant grace and presence of the almighty throughout different phases of the project and its successful completion.

We wish to express our sincere gratitude to our honorable Correspondent **Thiru.A.K.ILANGO B.Com., M.B.A., LLB.,** and other trust members for having provided us with all necessary infrastructures to undertake this project.

We extend our hearty gratitude to our honorable Principal **Dr.V.BALUSAMY B.E.(Hons)., MTech., Ph.D.,** for his consistent encouragement throughout our college days.

We would like to express our profound interest and sincere gratitude to our respected Head of the department **Dr.C.S.KANIMOZHI SELVI M.E., Ph.D.,** for her valuable guidance.

A special debt is owed to the project coordinator **Ms.S.SANTHIYA B.E., M.E.**. Assistant Professor, Department of Artificial Intelligence for her encouragement and valuable advice that made us carry out project work successfully.

We extend our sincere gratitude to our beloved guide **Ms.M.M.RAMYASRI B.E., M.E.,** Assistant Professor, Department of Artificial Intelligence for her ideas and suggestions, which have been very helpful to complete the project.

We are grateful to all the faculty and staff members of the Department of Artificial Intelligence and persons who directly and indirectly supported this project.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABELS

LIST OF ABBREVIATIONS

CNN	**:**	Convolutional Neural Networks
DBNs	**:**	Deep Belief Networks
LDA	**:**	Linear Discriminant Analysis
PCA	**:**	Principal Component Analysis
RNN	**:**	Recurrent Neural Networks

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Fingerprint identification is currently a prominent biometric authentication method, defined by its robustness, uniqueness, and ease of use. The technique has widespread applications in security systems, mobile phones, and forensic analysis. High classification accuracy is challenging to achieve, especially in the presence of small training sets, poor quality images, and complex feature representations. Traditional deep models require large datasets for effective generalization; with small data, the models can struggle to learn variability present in fingerprint patterns. These issues highlight the importance of advanced augmentation and feature optimization methods to push classification performance.

To counter the issue of fewer data being available, inversion and multi-augmentation methods are used for the generation of a training samples dataset in the present research. Augmentation methods such as rotation, scaling, flipping, and elastic deformation enhance the variability of the dataset and thus the model's generalization ability. In addition, inversion methods altering the polarity of the image make the feature extraction operation more robust. Overall, the methods help with fingerprint classification through the use of deep learning models trained on a larger dataset.

One of the most challenging issues in fingerprint classification is the large dimensionality of the extracted features, which could lead to increased computational cost and the presence of redundant information. To improve the feature representation, the dimensionality reduction techniques, such as Principal

Component Analysis (PCA) and Linear Discriminant Analysis (LDA), are employed. PCA drastically reduces redundant features without compromising essential information, whereas LDA optimizes class separability and thus improves classification efficiency. By concatenating the reduced feature maps, the system acquires a more compact and informative feature representation of fingerprint data, leading to quicker and more accurate classification. For feature extraction, the present work utilizes pretrained Convolutional Neural Networks (CNNs) such as AlexNet and GoogleNet, which are extremely effective for image recognition. In addition, complex architectures such as Recurrent Neural Networks (RNNs) and Deep Belief Networks (DBNs) are taken into consideration for improving the accuracy of classification. These models can learn sequential and spatial dependencies of fingerprint patterns and thus improve their ability to classify different categories. With the integration of data augmentation, feature reduction, and deep learning, the proposed approach maximizes fingerprint classification performance without compromising computational efficiency.

This work seeks to create a high-performance and scalable fingerprint recognition system to be applied in real-world applications such as security, law enforcement, and personal identification. By integrating augmentation and dimension reduction with state-of-the-art deep learning models, this work presents a well-established framework for enhancing fingerprint classification. The suggested method achieves the capability of biometric systems to function even in a situation with limited training data, which eventually leads to more efficient and reliable fingerprint-based verification.

1.2 OBJECTIVES

- Increase Data Availability – Implement inversion and multi-augmentation techniques for generating diverse training samples, thereby circumventing the limitation of scarce fingerprint data.
- Improve Feature Representation – Implement dimensionality reduction techniques such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) to improve feature selection and

combination and hence improve computational complexity.

- Improve Feature Extraction – Use pre-trained Convolutional Neural Networks (CNNs) such as AlexNet and GoogleNet to extract meaningful fingerprint features, thus enhancing classification efficiency.
- Visit Advanced Architectures – Research the employment of Recurrent Neural Networks (RNNs) and Deep Belief Networks (DBNs) to enhance the accuracy of fingerprint pattern classification and pattern recognition.
- Build a scalable architecture – Design a fingerprint classification system that is computationally efficient and scalable, making it practicable for use in real-life biometric authentication applications.
- Measure Model Effectiveness – Assess classification accuracy, precision, recall, and computational efficiency to provide the best performance in the context of fingerprint identification.

1.3 SCOPE

- Biometric Security Systems – The proposed fingerprint classification system can be integrated into security and authentication systems for cellular phones, financial applications, and access control systems.
- Forensic Identification – The technique can be used by the police to improve fingerprint matching in criminal investigations to facilitate identification even when there are incomplete or partial prints.
- Border Control and Immigration – The model can be utilized in automated immigration systems for passport checks and identification of passengers, thus enabling faster and more secure processing.
- Healthcare and Patient Identification – Fingerprint-based authentication can be utilized by hospitals and medical centers to securely access patient records and prevent identity fraud.

CHAPTER 2

SYSTEM ANALYSIS

2.1 LITERATURE REVIEW

Li et al. (2021) introduced a joint discriminative feature learning (JDFL) framework for multimodal finger recognition, combining finger vein (FV) and finger knuckle print (FKP) patterns. The proposed approach addresses the limitations of traditional methods, which separately extract features from different modalities and ignore correlations between them. The authors developed a feature learning algorithm that maximizes the distance between-class samples, minimizes the distance within-class samples, and enhances the correlation between modalities. Their method outperformed existing finger recognition techniques in terms of recognition performance, providing a more effective solution for multimodal biometric systems.

Li et al. (2021) proposed a discriminant local coding-based convolutional neural network (LC-CNN) for multimodal finger recognition, combining fingerprint, finger-vein, and finger-knuckle-print traits. The approach utilizes a weighted local gradient coding (WLGC) operator to enhance discriminative features, followed by a local coding layer (LC-layer) that is reconstructed using predefined convolution layers. The LC-CNN model extracts deep features from the tri-modal finger images, and the feature vectors are then classified using a Support Vector Machine (SVM). The experimental results showed that the proposed method achieved stable, highly accurate, and robust performance in multimodal finger recognition.

Abu-Faraj et al. (2021) conducted a comparative analysis of fingerprint feature extraction methods, focusing on techniques like K-means, Local Binary Pattern (LBP), Wavelet Packet Transform (WPT), and minutiae-based methods. The study aimed to assess the uniqueness and stability of extracted features, particularly under different rotation conditions. Their findings highlight the importance of stable features that remain consistent across various rotations, allowing the use of a single feature vector for fingerprint recognition. The analysis provides recommendations for experts in fingerprint recognition systems, emphasizing the need for methods that balance memory efficiency and classification speed.

Shaheed et al. (2021) conducted a systematic review on physiological-based biometric recognition systems, covering modalities like finger vein, palm vein, fingerprint, face, lips, iris, and retina. The authors evaluated various traditional and deep learning-based processing methods, focusing on the biometric steps of preprocessing, feature extraction, and classification. They highlighted the performance metrics, challenges, and future trends of both conventional and deep learning approaches. The review stressed the need for more robust physiological-based methods to improve the performance and efficiency of biometric systems, offering valuable insights for future research in the field.

Chowdhury and Imtiaz (2022) conducted a systematic review on contactless fingerprint recognition using deep learning. They discussed the limitations of traditional contact-based fingerprint systems and the advantages of contactless systems, particularly those utilizing deep learning methods, which showed higher accuracy. The review focused on three key aspects: finger photo capture methods and image sensors, classical image preprocessing for recognition tasks, and deep learning approaches for contactless fingerprint recognition. They identified eight relevant studies and concluded that while deep learning offers significant benefits for biometrics, there are still gaps that need addressing for real-world applications.

Hou, Zhang, and Yan (2022) reviewed the advancements in finger-vein biometric recognition, focusing on its four main steps: image acquisition, image preprocessing, feature extraction, and matching. They summarized various feature extraction methods, including template-based, representation-based, and learning-based approaches, with an emphasis on deep learning techniques. The review included case studies from finger-vein databases to demonstrate the effectiveness of these methods. The authors also discussed challenges and future prospects, particularly in 3-D finger-vein recognition, to guide future research directions in the field.

Rajasekar et al. (2022) proposed an enhanced multimodal biometric recognition approach for smart cities, leveraging score-level fusion and an optimized fuzzy genetic algorithm. The technique addresses the challenges of improving accuracy and recognition rates in biometric systems. Experimental results showed significant improvements over existing methods, with a high accuracy rate of 99.88% and a low equal error rate of 0.18%. The integration of a fuzzy strategy with soft computing techniques, such as the fuzzy genetic algorithm, played a key role in enhancing the performance, particularly in terms of false acceptance, false rejection, and overall accuracy.

Zhang et al. (2022) developed an innovative in-sensor reservoir computing (RC) system for latent fingerprint recognition, integrating deep ultraviolet (DUV) photo-synapses with a nonvolatile memristor array. The system addresses the inefficiencies of traditional fingerprint recognition methods, which suffer from the separation of sensor, memory, and processor. By leveraging the persistent photoconductivity (PPC) effect in GaOx photo-synapses, the proposed system can map complex input vectors into dimensionality-reduced output vectors. The in-sensor RC system achieved over 90% recognition accuracy for latent fingerprints, even under 15% stochastic noise. This work introduces a highly efficient, parallel in-memory computing system for fingerprint recognition, setting a new standard for latent fingerprint identification.

Minaee et al. (2023) conducted a comprehensive survey on deep learning-based biometric recognition, covering various modalities such as face, fingerprint, iris, palmprint, ear, voice, signature, and gait. The paper reviewed over 150 deep learning works, highlighting their effectiveness in improving accuracy across different biometric systems. It discussed the widely used datasets for each biometric, analyzed the performance of deep learning models on public benchmarks, and addressed the challenges faced in biometric recognition. The survey also explored potential future research directions in the field, emphasizing the increasing integration of deep learning to enhance biometric authentication and security systems.

Zhang et al. (2024) introduced a novel metasurface-based sensing strategy for Terahertz (THz) refractive sensing and fingerprint recognition. Their approach leverages surface waves (SWs), which offer long-range transmission, strong confinement, and high sensitivity to interfaces. The researchers demonstrated that the metasurface-excited SWs could simultaneously perform high-resolution fingerprint recognition and refractive sensing, achieving a sensitivity of 215.5°/RIU. The method enabled the resolution of multiple fingerprint details within a continuous spectrum, achieving phase changes up to 126.4° in the THz band. This breakthrough expands the potential of metasurface-excited SWs for ultrasensitive bio-sensing applications.

2.2 SUMMARY

Fingerprint classification has progressed from traditional minutiae-based and ridge pattern analysis to deep learning-driven methods that handle issues including noise, low image quality, and pattern variances. Convolutional Neural Networks (CNNs), such as AlexNet and GoogleNet, have considerably improved feature extraction, whilst advanced architectures like Recurrent Neural Networks (RNNs) and Deep Belief Networks (DBNs) improve spatial and sequential feature learning. Data augmentation techniques such as rotation, scaling, and inversion have been used to increase training datasets and improve generalization. Furthermore, dimensionality reduction methods such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) improve feature selection, lowering computing complexity while keeping crucial fingerprint properties.

CHAPTER 3

SYSTEM REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

CPU type	:	Ryzen-7 4600HS
Ram size	:	8 GB
Hard Disk Capacity	:	500 GB

3.2 SOFTWARE REQUIREMENTS

Operating System	:	Windows 10
Language	:	Python (version3)
Tool	:	Google Colab

3.3 SOFTWARE DESCRIPTION

3.3.1 Python

Python is essential for applying the approaches described in this paper, as it allows for efficient fingerprint classification via deep learning and data processing. Python tools such as TensorFlow and PyTorch make it easier to create and train Convolutional Neural Networks (CNNs) like AlexNet and GoogleNet, as well as sophisticated architectures like Recurrent Neural Networks (RNNs) and

Deep Belief Networks (DBNs). Preprocessing fingerprint images using OpenCV includes inversion as well as multi-augmentation methods including flipping, scaling, and rotation. Scikit-learn includes tools for dimensionality reduction approaches such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) to optimize feature selection. Furthermore, NumPy and Pandas help with fingerprint dataset processing and analysis, and Matplotlib and Seaborn depict classification performance using accuracy curves and confusion matrices, respectively. Python is perfect for combining deep learning, augmentation, and feature reduction to improve fingerprint classification efficiency and accuracy because of its adaptability and wide library support.

3.3.2 Google Colab Notebook

In this study, Google Colab Notebook is used to develop, train, and test deep learning models for fingerprint categorization. Given that it offers free access to GPUs and TPUs, it is perfect for training computationally demanding models such as RNNs, DBNs, GoogleNet, and AlexNet. For dimensionality reduction, feature extraction, and augmentation, Python libraries like TensorFlow, PyTorch, OpenCV, and Scikit-learn are utilized. Colab's interface with Google Drive enables simple dataset administration, while its interactive environment facilitates real-time visualization of accuracy, loss curves, and confusion matrices, allowing for quick model evaluation and experimentation.

CHAPTER 4

PROPOSED SYSTEM

4.1 SYSTEM ARCHITECTURE

The system proposed here will enhance fingerprint classification by applying data augmentation, dimensionality reduction, and deep learning models for better accuracy and scalability. The majority of traditional fingerprint recognition systems are plagued by the limitations of small data and high computational costs, which degrade model efficiency. To avoid these limitations, the proposed system employs advanced augmentation strategies, such as inversion, rotation, and scaling, to generate extra training data. This leads to a diversified dataset, which enables the model to generalize efficiently for different fingerprint patterns and variations.

To improve fingerprint classification optimization, dimensionality reduction techniques like Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are used. These techniques help optimize the derived feature maps by eliminating redundant information and focusing on the most significant features of fingerprints. The combination of optimized feature maps enhances the efficiency of classification while, at the same time, reducing computational complexity and therefore enhancing the system's adaptability for implementation.

The architecture employs pretrained Convolutional Neural Networks (CNNs) such as AlexNet and GoogleNet to perform automatic feature learning, taking advantage of their deep hierarchical structures to extract rich fingerprint features efficiently. Furthermore, more sophisticated architectures such as Recurrent Neural Networks (RNNs) and Deep Belief Networks (DBNs) are investigated to improve pattern detection capability and improve classification accuracy. The models are integrated into a hybrid strategy that reconciles feature learning, model interpretability, and computational complexity.

In short, the proposed framework is designed to develop a strong, scalable, and high-speed fingerprint classification system that can be utilized in a wide range of biometric security and authentication systems. By incorporating augmentation methods, dimensionality reduction operations, and deep learning techniques, the framework greatly improves fingerprint recognition, enhances generalization ability, and offers high levels of classifying accuracy even with small training sets. Fig 4.1 shows the workflow of the project.

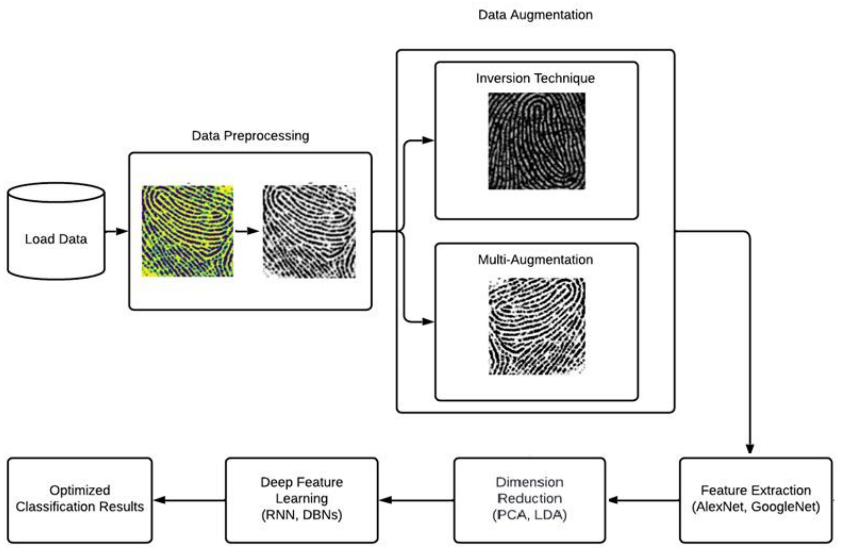

Data Augmentation
Inversion Technique
Data Preprocessing
Load Data
Multi-Augmentation
Optimized Classification Results
Deep Feature Learning (RNN, DBNs)
Dimension Reduction (PCA, LDA)
Feature Extraction (AlexNet, GoogleNet)

Fig 4.1 Proposed System

4.2 MODULE DESCRIPTION

4.2.1 Dataset Collection

The FVC2000_DB4_B dataset is a structured fingerprint dataset with photos organized into folders called np_data, train_data, and real_data. The train_data folder contains around 800 fingerprint images with filenames written as "XXXXX_YY", where XXXXX is the finger index and YY is the user ID. The collection contains ten separate fingerprint classifications for both left and right-hand fingers. Each finger type is labeled from 0 to 9, with thumbs, index, middle, ring, and little fingers included. This dataset is useful for fingerprint classification and biometric authentication research, as it contains a wide range of fingerprint patterns for model training and evaluation. The below Fig 4.2 shows the fingerprint image.

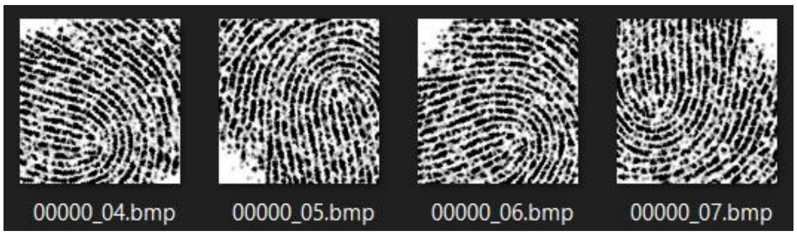


Fig 4.2 Sample fingerprint image

4.2.2 Data pre-processing

Fingerprint images are also prone to noise, distortions, and inconsistency in contrast, which can affect the accuracy of classification. To reduce such issues, different preprocessing methods are used to enhance fingerprint patterns and enable better feature extraction.

4.2.2.1 Grayscale Conversion

The fingerprint images are transformed to grayscale to remove irrelevant color information and highlight textural information and ridge patterns. Since fingerprint recognition relies on ridge-valley structures rather than color information, the conversion to grayscale simplifies the input data without reducing the computation required. The operation is performed by (4.1):

$$I = 0.2989R + 0.5870G + 0.1140B \tag{4.1}$$

where R,G,B are red, green, and blue intensities, respectively. The grayscale image obtained enhances ridges' contrast, which is better suited for feature extraction.

4.2.2.2 Resizing

Fingerprint images are resized to the same size of 256×256 pixels in order to be consistent with the dataset. The process ensures the images are uniform in size, thus preventing variance due to variations in image sizes. Uniform image size also helps the model to process the inputs in a consistent manner, avoiding overfitting or misclassification.

4.2.2.3 Gaussian Blur for Denoising

Gaussian blur is applied to improve the smoothness of fingerprint images and reduce noise, without compromising major ridge structures. The filtering technique is very effective in removing small distortions and unnecessary variations, thus improving the readability of fingerprint patterns. The Gaussian filter (4.2) is defined mathematically as:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4.2}$$

where σ controls the amount of blurring. A 3×3 Gaussian filter is used to enhance the fingerprint patterns without losing important details.

4.2.2.4 Contrast Enhancement

Histogram equalization is used for the purpose of maximizing the contrast of fingerprint images to make ridge patterns more apparent. Pixel intensities in the image are redistributed by the process such that dark ridges and light valleys are well defined. The calculation is performed by the transformation function (4.3):

$$s_k = (L-1)\sum_{j=0}^{k} \frac{p_j}{N} \tag{4.3}$$

where L is the number of gray levels. P is the count of pixels with intensity. N is the number of pixels. This method improves fingerprints, particularly in poor-quality images, to increase the accuracy of classification. Fig 4.3 shows the fingerprint image after preprocessing.

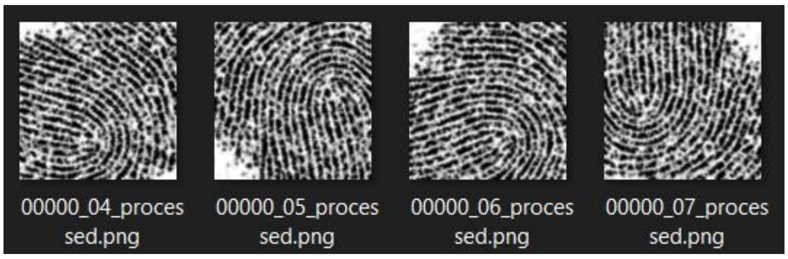


Fig 4.3 Preprocessed Fingerprint Image

4.2.3 Data augmentation

Augmentation is crucial to fingerprint classification for improving the robustness and generalizability of machine learning models. As fingerprint data sets are typically small, techniques of augmentation generate variations in data, thus preventing model overfitting and improving performance on out-of-sample points. Inversion-based augmentation, in specific, can impose novel transformations which maintain fingerprint pattern and mimic variations in real-world environments due to diverse scanning conditions, skin folds, and sensor noise. Augmentation ensures the model learns good fingerprint classification under minimal structural transformations.

4.2.3.1 Inversion Augmentation

The process of enhancement done in the code is the inversion of the fingerprint images by vertical flipping of segments of the original image. Initially, each fingerprint image is split into two equal halves along the vertical axis. Mathematically, if the dimensions of an image are (4.4):

$$I \in \mathbb{R}^{h \times w} \tag{4.4}$$

Both halves are subsequently rotated vertically by the following transformation (4.5):

$$I' = \text{flip}(I, \text{axis} = 0) \tag{4.5}$$

The enhanced images are derived by combining these reversed halves with their respective original halves. The two resulting forms are (4.6):

$$I_1 = \begin{bmatrix} I_{\text{top}} \\ \text{flip}(I_{\text{top}}, \text{axis} = 0) \end{bmatrix}, \quad I_2 = \begin{bmatrix} I_{\text{bottom}} \\ \text{flip}(I_{\text{bottom}}, \text{axis} = 0) \end{bmatrix} \tag{4.6}$$

Each freshly created fingerprint image is stored with a filename in the following format:

finger_index_user_id_inversion_id.png

In addition, metadata with the corresponding labels and information about inversion is retained in a text file to ensure traceability and consistency in subsequent analysis. The finalized dataset is serialized and saved as a pickle file to facilitate easy loading during model training. The output image of inversion technique is displayed in Fig 4.4.

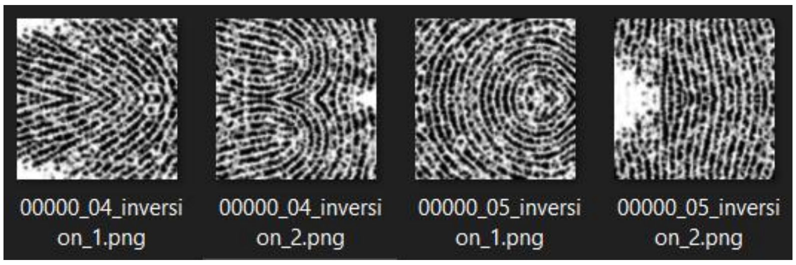


Fig 4.4 Inversion Augmentation result

4.2.3.2 Multi-Augmentation

Multi-augmentation is a method that uses multiple transformations on an image to increase dataset diversity. The transformations are used to make the model more robust by mimicking different real-world variations.

(i) Rotation and Cropping

All images are rotated ±45° prior to cropping to an equal size.

Counterclockwise Rotation (-45°) (4.7):

$$I' = R_{-45^\circ} \cdot I \tag{4.7}$$

Right Rotation (+45°) (4.8):

$$I' = R_{+45^\circ} \cdot I \tag{4.8}$$

(ii) Division into Halves

The diagram is split into two horizontal sections to show differences in partial fingerprints.

Top Half Extraction (4.9):

$$I_{\text{top}} = I[0 : h/2, :] \tag{4.9}$$

Bottom Half Extraction (4.10):

$$I_{\text{bottom}} = I[h/2 : h, :] \tag{4.10}$$

(iii) Rotated Halves Sum

Also, the rotated pictures are divided into halves to form more variations.

Left-Rotated Upper Half (4.11):

$$I_{\text{upper-left}} = R_{-45^\circ} \cdot I_{\text{top}} \tag{4.11}$$

Left-Rotated Lower Hemisphere (4.12):

$$I_{\text{lower-left}} = R_{-45^\circ} \cdot I_{\text{bottom}} \tag{4.12}$$

Right-Rotated Upper Hemisphere(4.13):

$$I_{\text{upper-right}} = R_{+45^\circ} \cdot I_{\text{top}} \tag{4.13}$$

Right-Rotated Lower Portion(4.14):

$$I_{\text{lower-right}} = R_{+45^\circ} \cdot I_{\text{bottom}} \tag{4.14}$$

Augmented Image Storage Format Each of the improved fingerprint images is stored systematically with a structured filename comprising:

finger_index_user_id_multiple_augmentation_identifier.png

All metadata of the images (user ID, finger index, augmentation type) are also saved in a structured text file (multi_augmented_labels.txt) and a pickle file (multi_augmented_fingerprint_dataset.pkl) for convenient access and model training. The output image of multi-augmented technique is displayed in Fig 4.5.

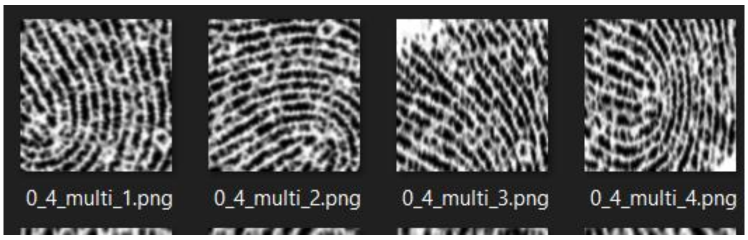


Fig 4.5 Multi-Augmented result

4.2.4 Feature Extraction

Feature extraction is a critical step in fingerprint recognition because it enables raw image data to be converted into useful representations that enable efficient classification and matching. Because fingerprint patterns are complex, identification of unique features maximizes model efficiency while at the same time reducing computational cost. In the current research, feature extraction is applied to determine major fingerprint features to guarantee deep learning models focus on informative structural features rather than processing entire images. Using pre-trained convolutional neural networks (CNNs), extracted features are employed as high-dimensional representations that retain essential information for identity verification and authentication.

4.2.4.1 AlexNet

AlexNet is chosen for feature extraction due to its deep architecture and proven success in image classification. Its convolutional and pooling layers effectively extract complex spatial hierarchies in fingerprint images, from low-level edges to high-level ridge patterns. This optimized feature space reduces the need for handcrafted feature engineering, leading to strong fingerprint representation and improved recognition accuracy.To enhance fingerprint identification, inverted images and multi-augmented images are used as input for feature extraction. This results in two feature sets: AlexNet-Inverted Features, AlexNet-Multi-Augmented Features. These extracted features retain key fingerprint details while maximizing diversity, improving generalization, and boosting recognition accuracy. Fig 4.6 displays the architecture of AlexNet for feature extraction. Fig 4.7 shows the summary of AlexNet architecture.

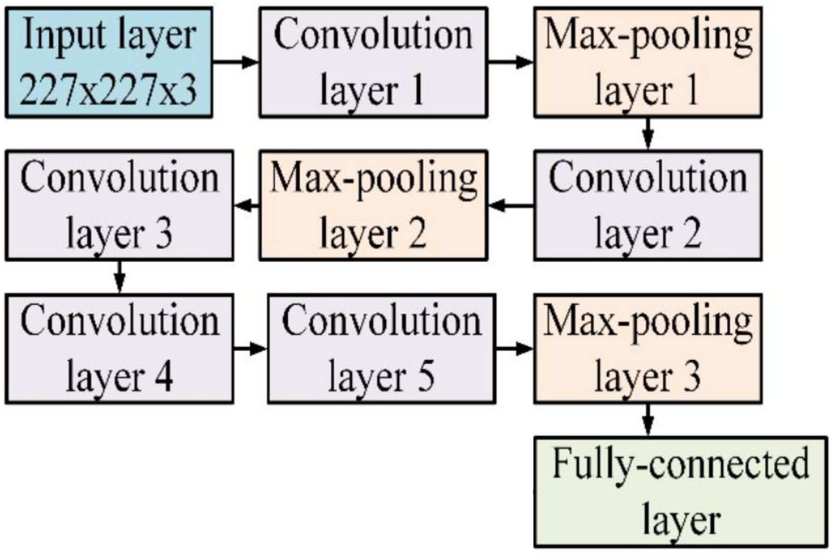
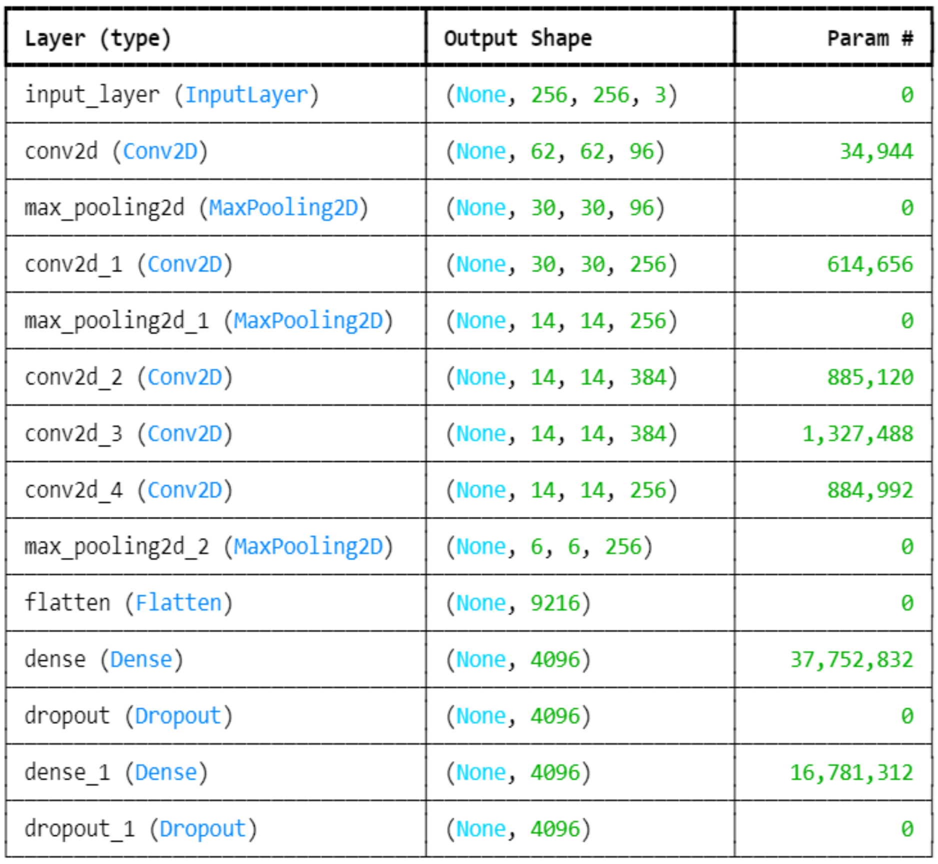


Fig 4.6 AlexNet Architecture



Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 256, 256, 3)	0
conv2d (Conv2D)	(None, 62, 62, 96)	34,944
max_pooling2d (MaxPooling2D)	(None, 30, 30, 96)	0
conv2d_1 (Conv2D)	(None, 30, 30, 256)	614,656
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 256)	0
conv2d_2 (Conv2D)	(None, 14, 14, 384)	885,120
conv2d_3 (Conv2D)	(None, 14, 14, 384)	1,327,488
conv2d_4 (Conv2D)	(None, 14, 14, 256)	884,992
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 256)	0
flatten (Flatten)	(None, 9216)	0
dense (Dense)	(None, 4096)	37,752,832
dropout (Dropout)	(None, 4096)	0
dense_1 (Dense)	(None, 4096)	16,781,312
dropout_1 (Dropout)	(None, 4096)	0

Fig 4.7 Summary of AlexNet Architecture

4.2.4.2 GoogleNet

GoogleNet, specifically its InceptionV3 model, offers an alternative feature extraction approach. Its Inception module captures multi-scale features from a single layer, making it effective for processing fingerprint patterns with varying orientations and complexities. With a deep architecture and strong parameter tuning, GoogleNet efficiently extracts fingerprint details with low computational cost. The diverse and complementary features obtained from InceptionV3 further enhance recognition performance.Like AlexNet, GoogleNet also processes inverted images and multi-augmented images to generate two feature sets:GoogleNet-Inverted Features, GoogleNet-Multi-Augmented Features. These extracted features capture essential fingerprint traits while ensuring diversity. Fig 4.8 displays the architecture of GoogleNet for feature extraction.

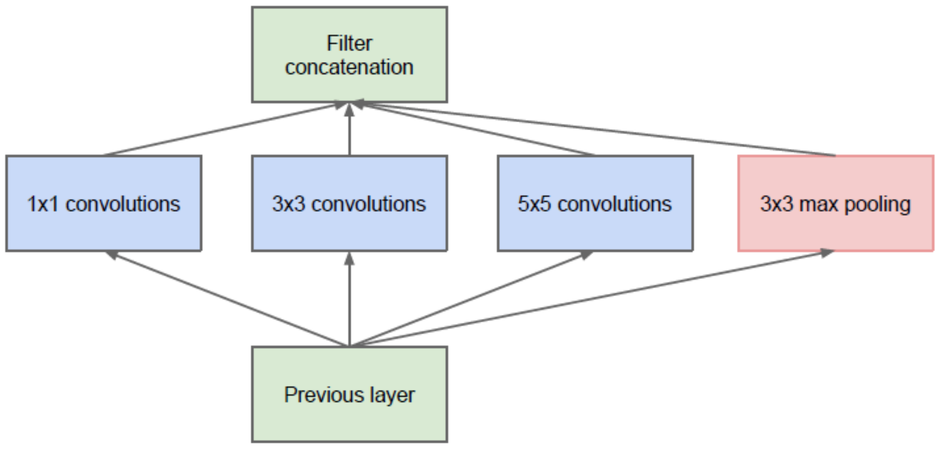


Fig 4.8 GoogleNet Architecture

4.2.5 Dimensionality Reduction

Dimensionality reduction is used to decrease high-dimensional fingerprint features without losing critical information. PCA and LDA are utilized in this research to improve processing speed and classification performance. Since four feature sets are obtained from AlexNet and GoogleNet, respectively, each of the feature sets is then used with PCA and LDA, resulting in eight different final feature representations**.**

4.2.5.1 PCA

PCA is an unsupervised dimensionality reduction algorithm that maps high-dimensional data onto a lower-dimensional space by discovering the principal components directions in which data variance is the largest. Projection in this method reduces redundancy and noise while maintaining as much information as possible. Mathematically, the process of PCA construction is to calculate the covariance matrix of the data and then calculate the eigenvalue decomposition to find the principal components. For any dataset X with n samples and d dimensions, PCA accomplishes the following:

Step 1: Normalize the data into unit variance and zero mean.

Step 2: Compute the covariance matrix (4.15):

$$C = \frac{1}{n} X^T X \tag{4.15}$$

Step 3: Perform eigenvalue decomposition (4.16):

$$Cv = \lambda v \tag{4.16}$$

Where v are the eigenvectors (principal components) and λ are the eigenvalues.

Step 4: Select the top-ranked k eigenvectors of largest eigenvalues, which form the transformation matrix together.

Step 5: Project the original dataset to the lower-dimensionality space (4.17):

$$X' = XV_k \tag{4.17}$$

PCA is used in this research on every set of fingerprint features maintaining 95% variance in order to ensure necessary fingerprint information is retained and dimensionality is reduced. PCA feature sets after transformation are stored independently to be utilized in the future classification.

4.2.5.2 LDA

LDA is a supervised dimensionality reduction technique that attempts to optimize class separability. Unlike PCA, which optimizes variance, LDA attempts to discover a lower-dimensional space that maximizes class discrimination. This makes LDA especially suitable for fingerprint recognition, where precise classification is critical. The mathematical representation of LDA is to calculate the within-class scatter matrix and between-class scatter matrix.

Step 1: Compute the mean vector for each class (4.18):

$$m_i = \frac{1}{N_i} \sum_{x \in C_i} x \tag{4.18}$$

Step 2: Calculate the within-class scatter matrix (4..19):

$$S_W = \sum_{i=1}^{c} \sum_{x \in C_i} (x - m_i)(x - m_i)^T \tag{4.19}$$

Step 3: Calculate the between-class scatter matrix (4.20):

$$S_B = \sum_{i=1}^{c} N_i (m_i - m)(m_i - m)^T \tag{4.20}$$

Step 4: Solve the generalized eigenvalue problem (4.21):

$$S_W^{-1} S_B v = \lambda v \tag{4.21}$$

Step 5: Project the original dataset to the lower-dimensionality space (4.22):

$$X' = X V_k \tag{4.22}$$

v refers to eigenvectors (directions of discrimination). Select the appropriate k eigenvectors to construct the transformation matrix and project the data.

In this research, LDA is independently used with each of the four sets of extracted features such that the projected features maximize class discrimination between various fingerprint classes. The number of LDA components is fixed as the minimum between (number of classes - 1) or feature dimension. This preserves the best class separability. All four of the feature sets that are obtained are processed under PCA and LDA, with a total of eight feature sets being reduced. PCA is beneficial in preserving the most significant variations in the data, whereas LDA is targeted at maximizing separability between the classes. Transformed feature sets are then maintained for further processing, lowering computational complexity as well as classification accuracy.

4.2.6 Classification

Classification is one of the primary processes in this project because it enables accurate recognition of fingerprints on the basis of their derived characteristics. By classifying fingerprint samples into particular classes, the model can recognize useful patterns, enhancing safe means of verification and authentication. Classification as a process plays a central role in recognizing fingerprints appropriately, in which every fingerprint is assigned the proper category. Classification enhances the overall reliability and security of biometric systems as well, which makes it a central component of the project.

4.2.6.1 RNN

The Recurrent Neural Network (RNN) model employed in this study employs a Bidirectional Long Short-Term Memory (Bi-LSTM) architecture for fingerprint classification. The model comprises several levels of Bi-LSTM layers augmented with added dropout and batch normalization to enhance generalization and prevent the risks of overfitting. The last dense layer incorporates a softmax activation function to classify fingerprints into varied classes. For efficient training, the AdamW optimizer is employed, while early stopping augmented with learning rate reduction is used to attain the best model performance. The results, such as measures of accuracy and loss, are graphed to track training progress, and the model is stored for future use. The RNN processes eight different models obtained from techniques of feature extraction and dimension reduction. Specifically, all four sets of features obtained from fingerprint images are subjected to both Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA), leading to eight different sets of features. These reduced sets of features support the optimization of classification performance by preserving key fingerprint features while removing redundant information. RNN architecture is shown in Fig 4.9 shows the result of multiple models.

4.2.6.2 DBNs

The model adopted in this research study, based on the Deep Belief Network (DBN), is a feedforward neural network structure tailored to be applied for fingerprint classification. It involves the application of multiple dense layers with the incorporation of dropout and L2 regularization techniques to ensure generalization capabilities and prevent the risk of overfitting. Training of the model is achieved through the use of the Adam optimizer with categorical cross-entropy loss, and learning rate scheduling and early stopping techniques are employed to increase training efficiency. The classification final layer incorporates the softmax activation function to achieve effective fingerprint categorization. The dataset consists of fingerprint feature sets extracted through Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). Eight distinct feature sets are handled in this approach. The reduced feature sets

are important for optimizing classification efficiency by retaining key fingerprint

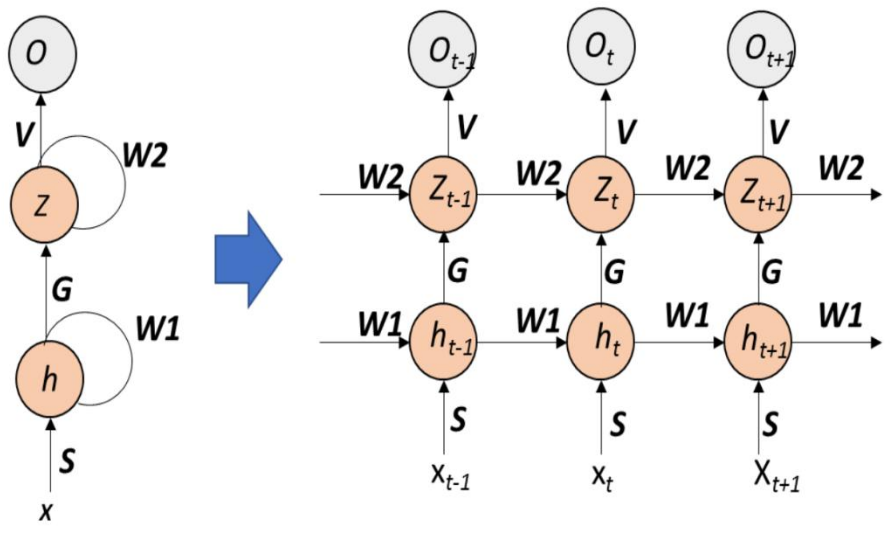


Fig 4.9 RNN Architecture

features and removing redundant information. The performance of the model is measured using accuracy metrics, confusion matrices, and classification reports, thereby ensuring efficient fingerprint sample classification. DBNs architecture is shown in Fig 4.10 shows the result of multiple models.

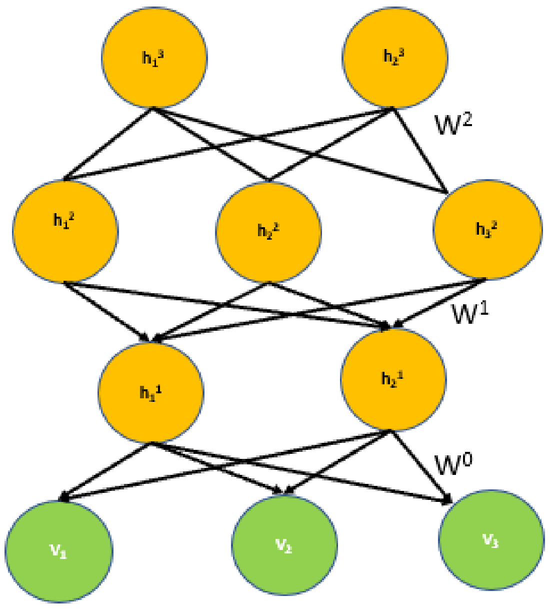


Fig 4.10 DBNs Architecture

4.3 VISUALIZATIONS

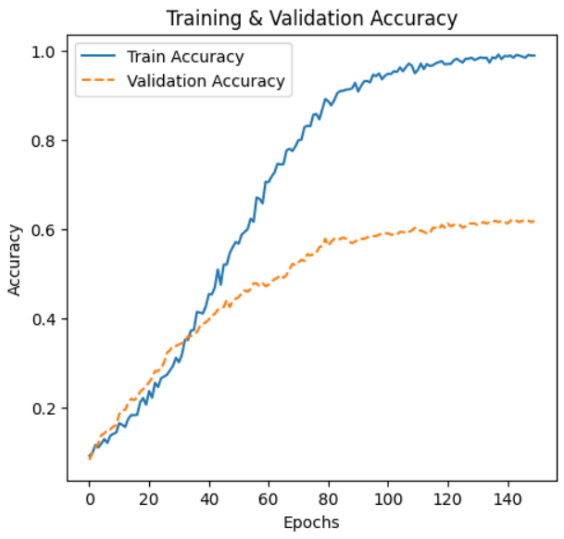


Fig 4.11 Accuracy of RNN with Inverted AlexNet PCA

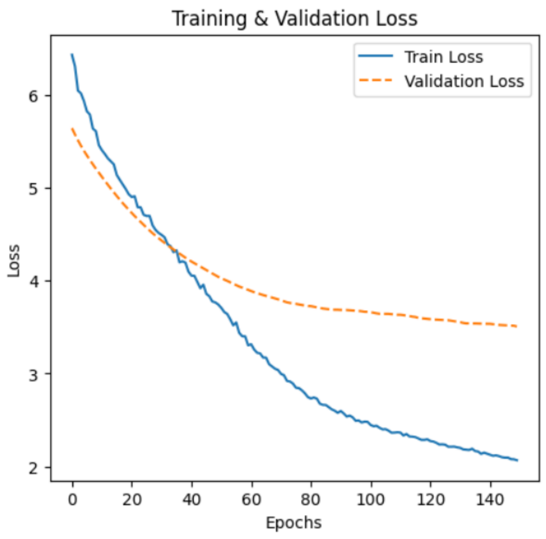


Fig 4.12 Loss of RNN with Inverted AlexNet PCA

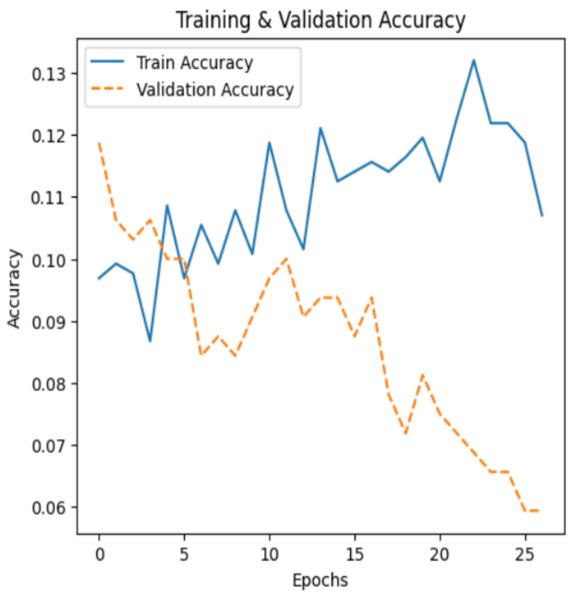


Fig 4.13 Accuracy of RNN with Inverted AlexNet LDA

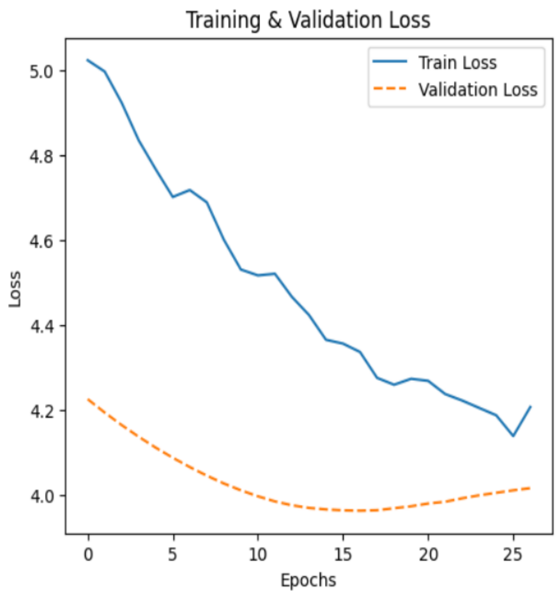


Fig 4.14 Loss of RNN with Inverted AlexNet LDA

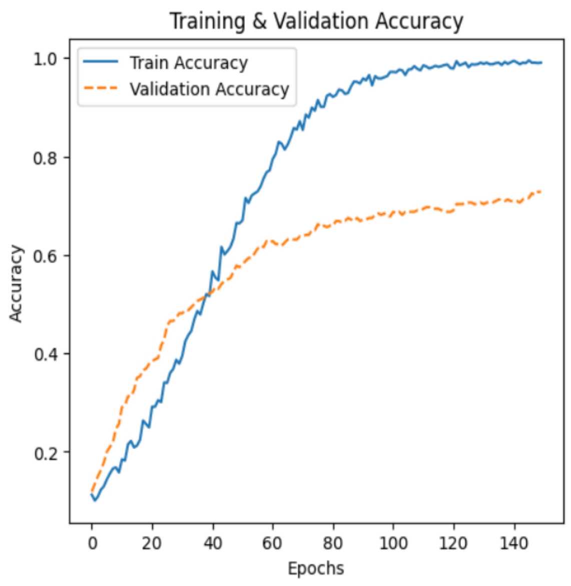


Fig 4.15 Accuracy of RNN with Inverted GoogleNetNet PCA

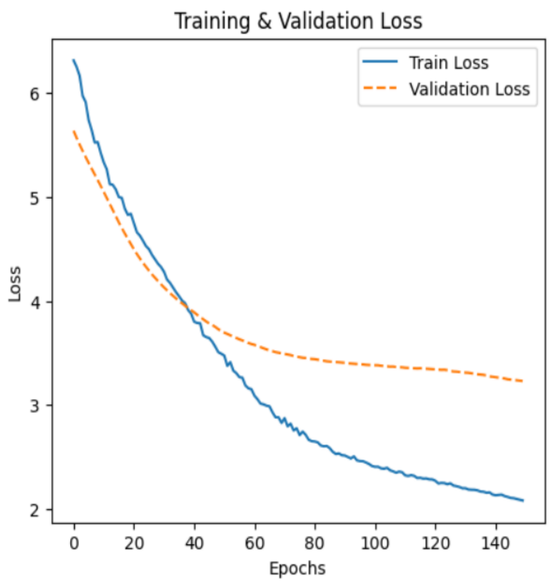


Fig 4.16 Loss of RNN with Inverted GoogleNet PCA

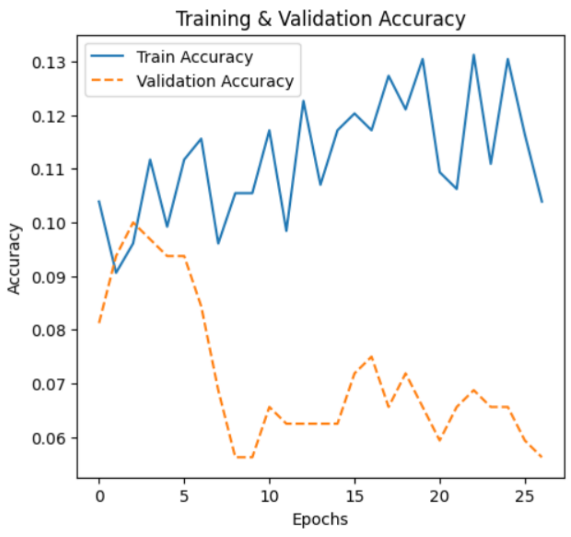


Fig 4.17 Accuracy of RNN with Inverted GoogleNetNet LDA

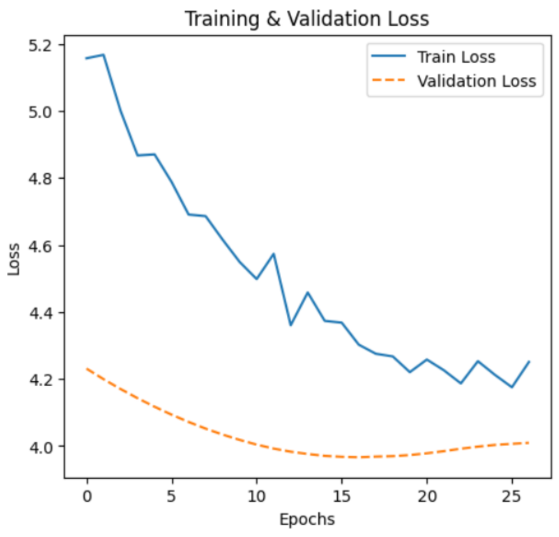


Fig 4.18 Loss of RNN with Inverted GoogleNetNet LDA

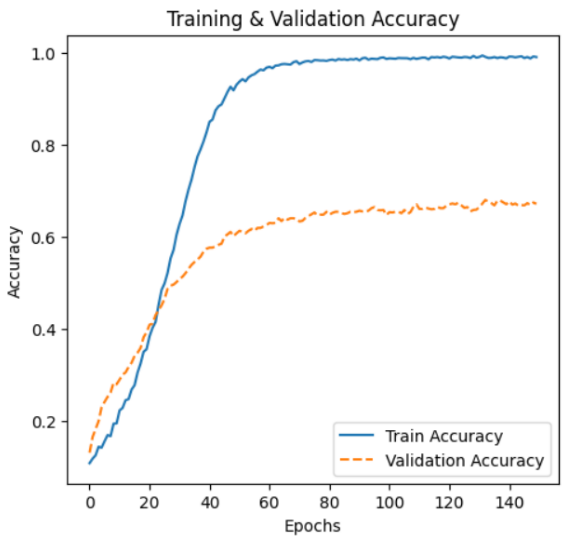


Fig 4.19 Accuracy of RNN with Multi-Augmented AlexNet PCA

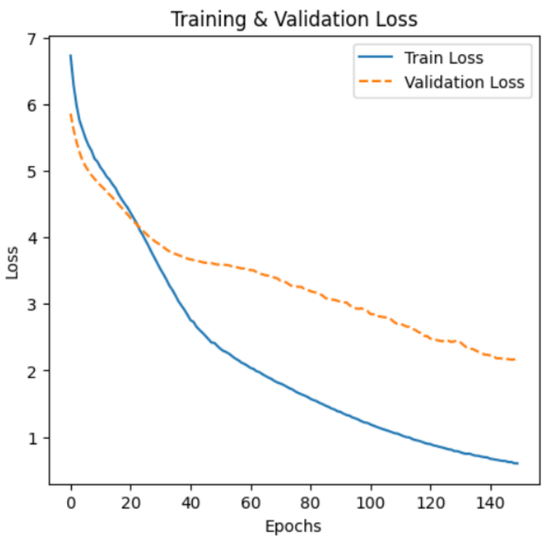
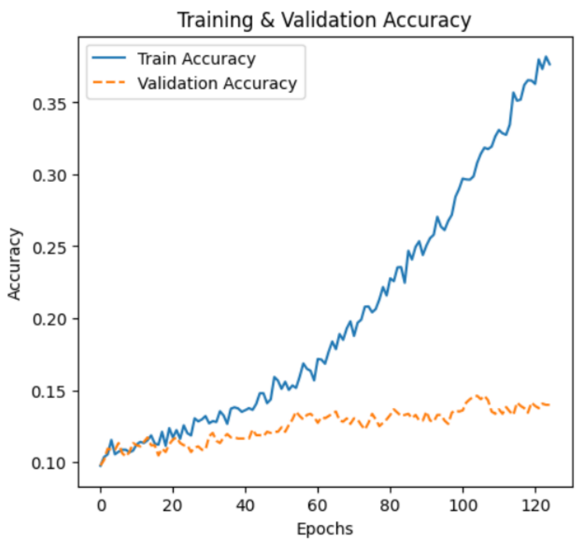


Fig 4.20 Loss of RNN with Multi-Augmented AlexNet PCA



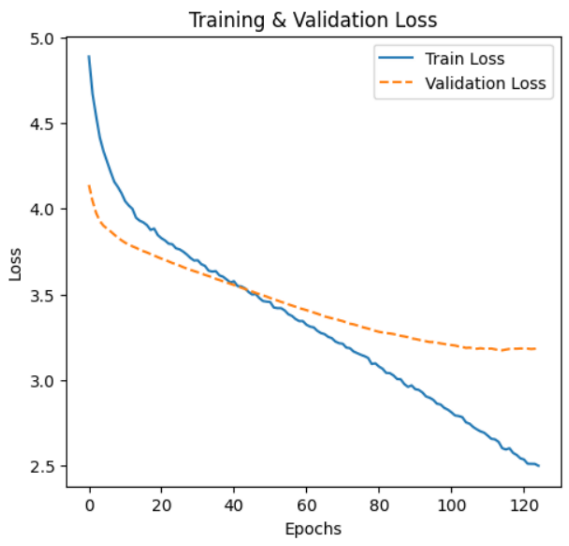
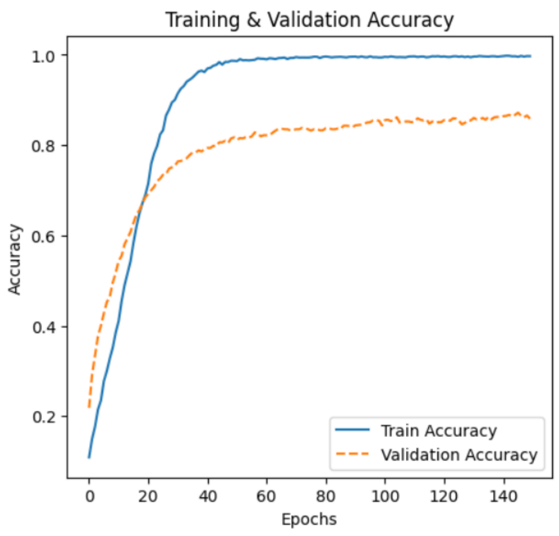


Fig 4.21 Accuracy of RNN with Multi-Augmented AlexNet LDA

Fig 4.22 Loss of RNN with Multi-Augmented AlexNet LDA



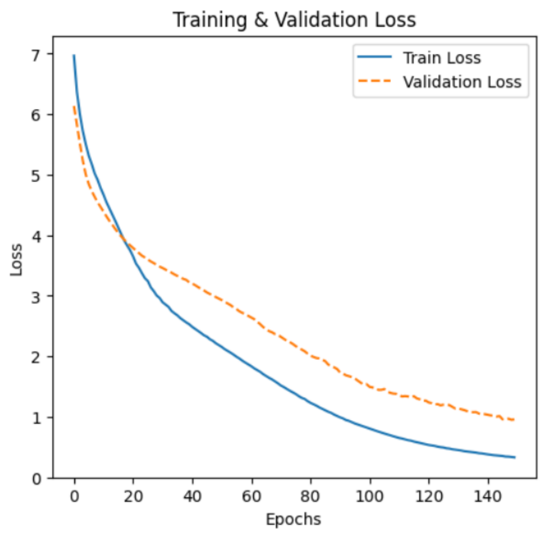
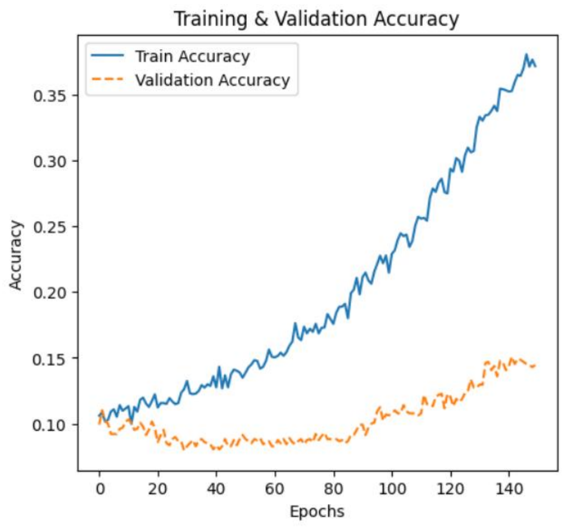


Fig 4.23 Accuracy of RNN with Multi-Augmented GoogleNetNet PCA

Fig 4.24 Loss of RNN with Multi-Augmented GoogleNetNet PCA



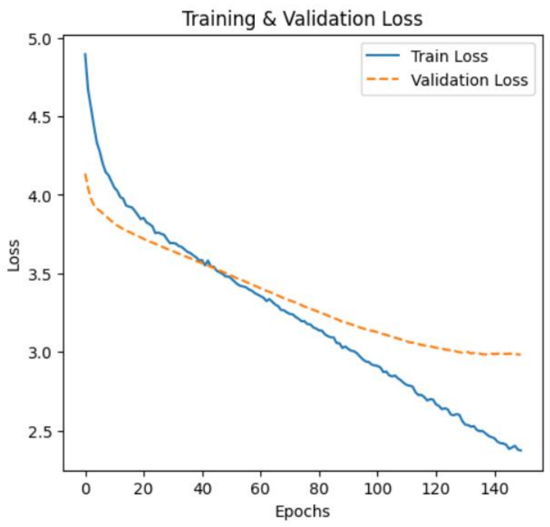
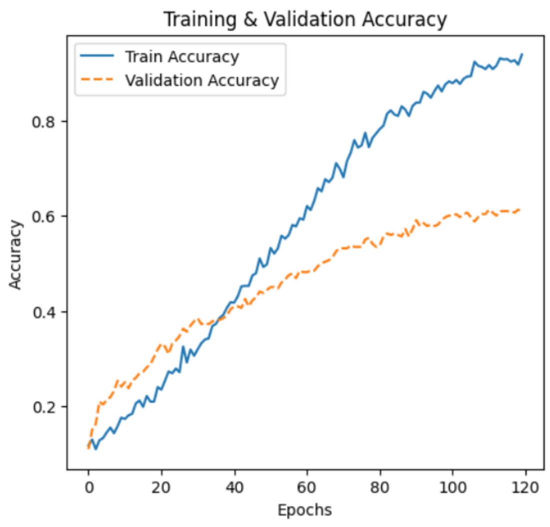


Fig 4.25 Accuracy of RNN with Multi-Augmented GoogleNetNet LDA

Fig 4.26 Loss of RNN with Multi-Augmented GoogleNetNet LDA



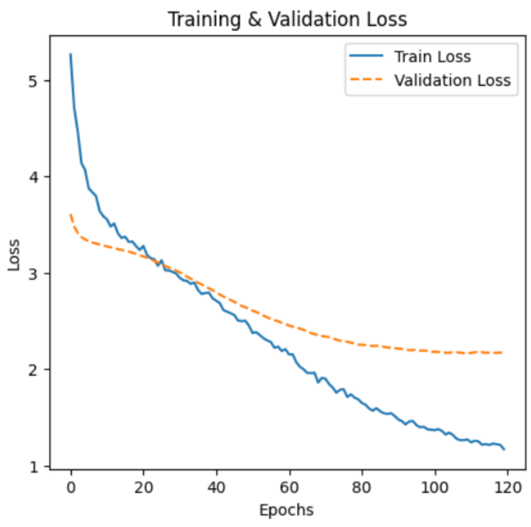


Fig 4.27 Accuracy of DBNs with Inverted AlexNet PCA

Fig 4.28 Loss of DBNs with Inverted AlexNet PCA

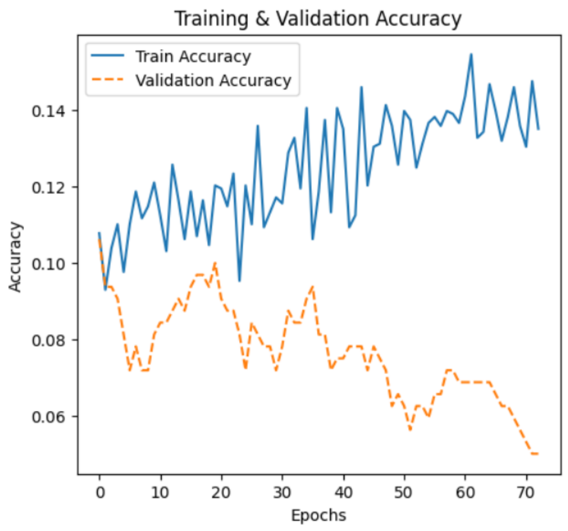


Fig 4.29 Accuracy of DBNs with Inverted AlexNet LDA

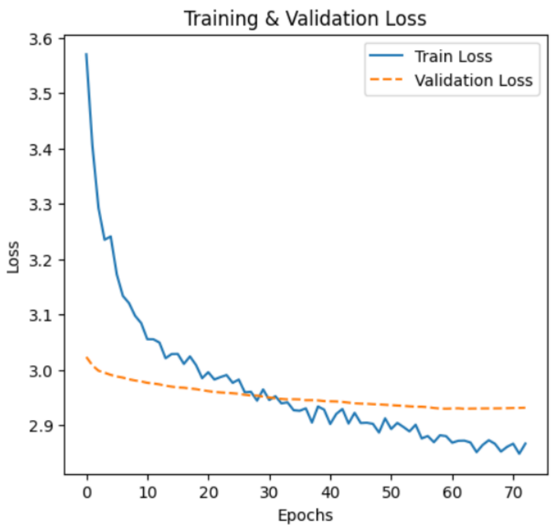


Fig 4.30 Loss of DBNs with Inverted AlexNet LDA

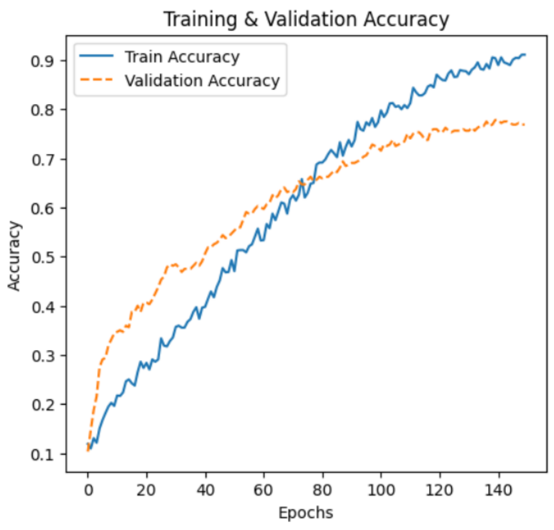


Fig 4.31 Accuracy of DBNs with Inverted GoogleNetNet PCA

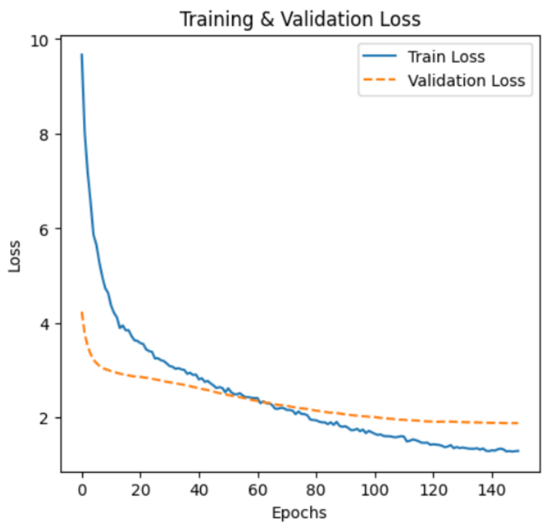


Fig 4.32 Loss of DBNs with Inverted GoogleNetNet PCA

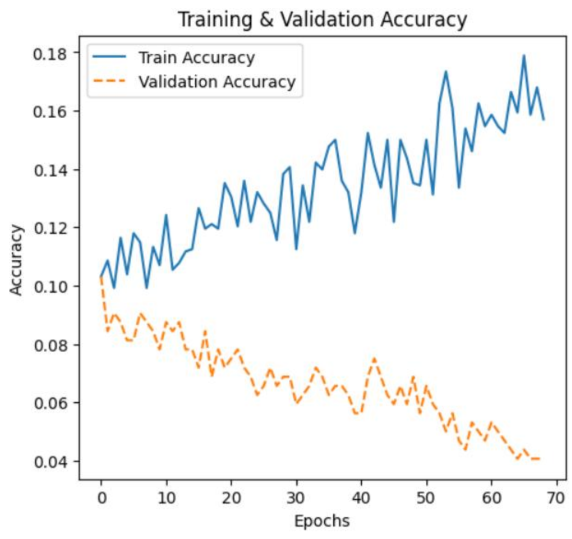


Fig 4.33 Accuracy of DBNs with GoogleNetNet AlexNet LDA

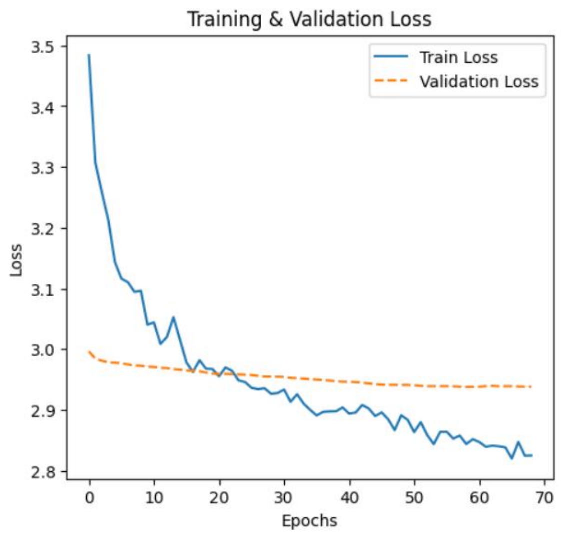


Fig 4.34 Loss of DBNs with Inverted GoogleNetNet LDA

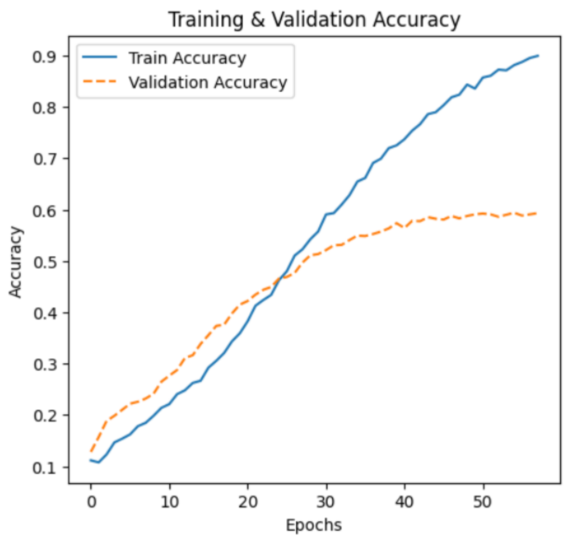


Fig 4.35 Accuracy of DBNs with Multi-Augmented AlexNet PCA

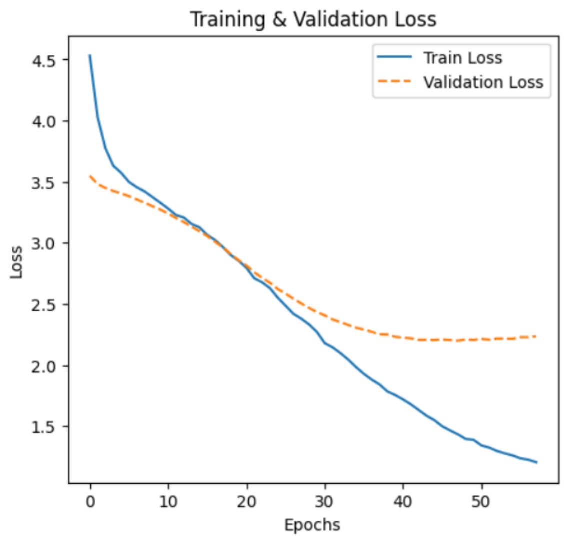
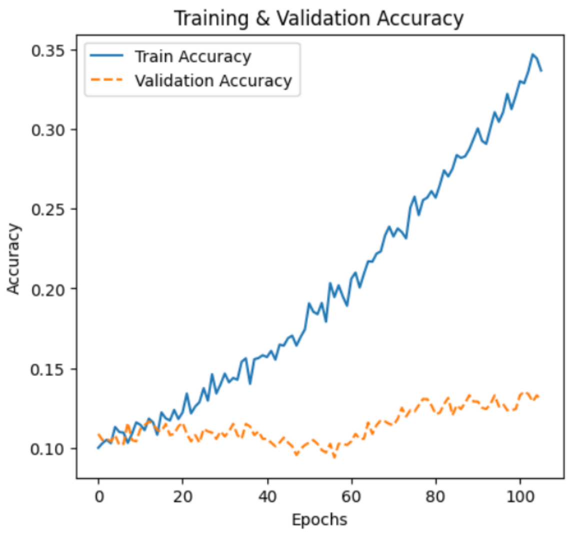


Fig 4.36 Loss of DBNs with Multi-Augmented AlexNet PCA



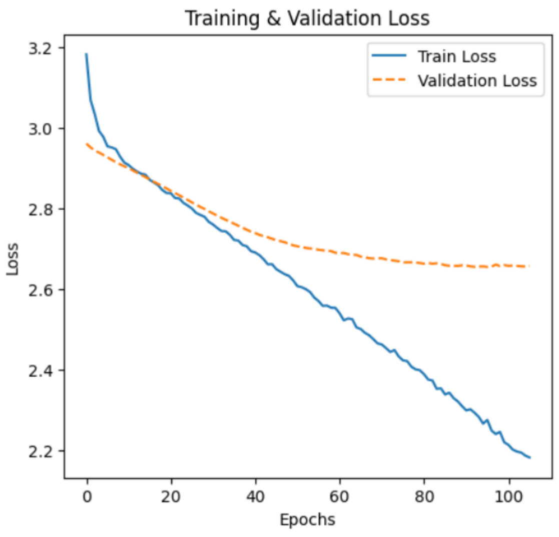
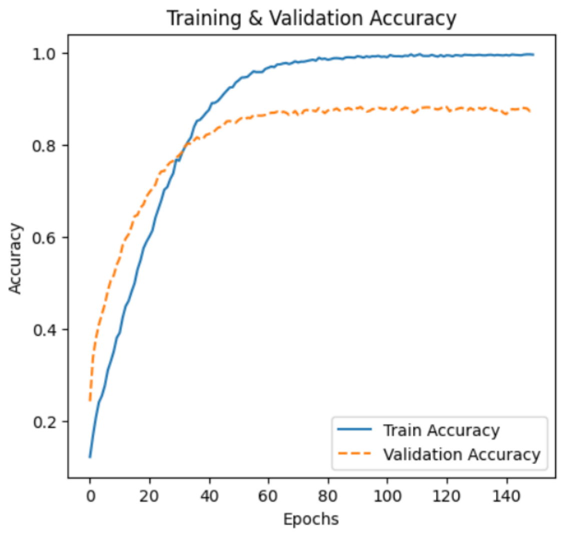


Fig 4.37 Accuracy of DBNs with Multi-Augmented AlexNet LDA

Fig 4.38 Loss of DBNs with Multi-Augmented AlexNet LDA



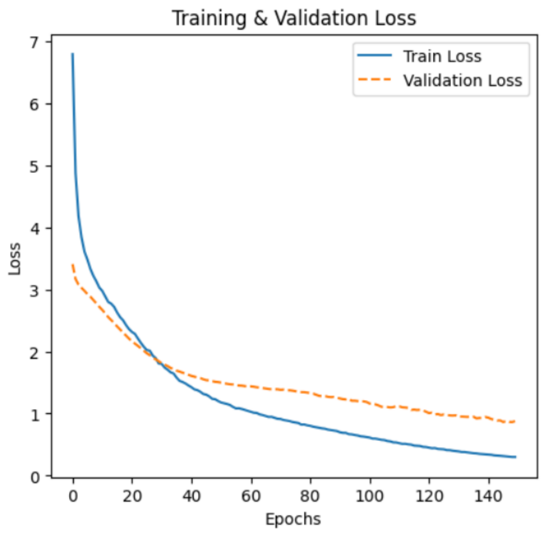
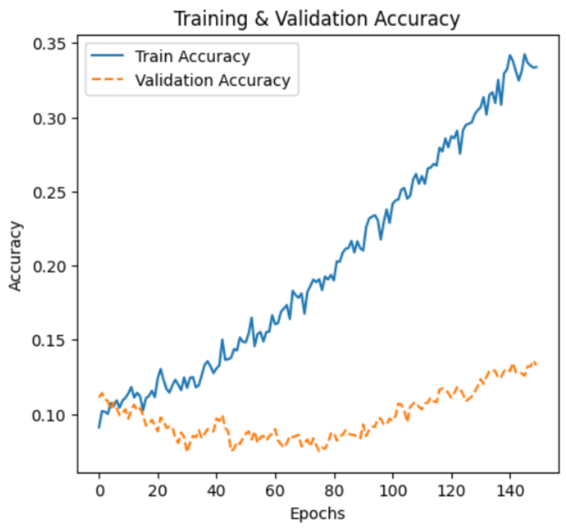


Fig 4.39 Accuracy of DBNs with Multi-Augmented GoogleNetNet PCA

Fig 4.40 Loss of DBNs with Multi-Augmented GoogleNetNet PCA



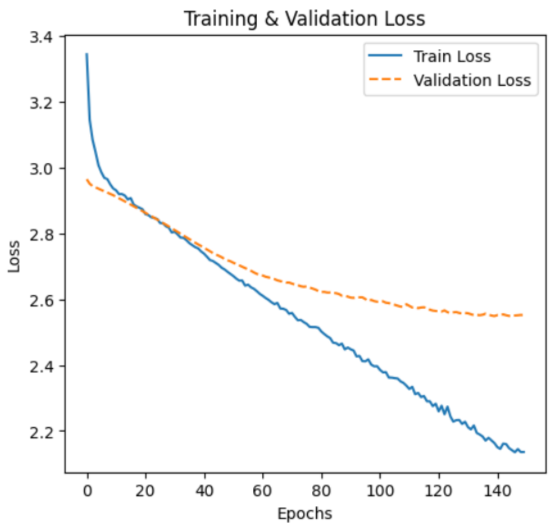


Fig 4.41 Accuracy of DBNs with Multi-Augmented GoogleNetNet LDA

Fig 4.42 Loss of DBNs with Multi-Augmented GoogleNetNet LDA

CHAPTER 5

PERFORMANCE ANALYSIS

1. Accuracy

Accuracy is a measure of the number of correct predictions out of all the predictions. It is defined in (5.1):

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.1}$$

where

True Positives (TP): Accurately forecasted positive cases.

True Negatives (TN): Properly predicted negative samples.

False Positives (FP): Erroneous predicted positive observations.

False Negatives (FN): Incorrectly classified negative samples.

2. Loss (Categorical Cross-Entropy)

The loss function used is categorical cross-entropy, which is a measure of the goodness of the predicted probability distribution in estimating the true distribution. It is defined as (5.2):

$$L = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j} \log(\hat{y}_{i,j}) \tag{5.2}$$

3. Confusion Matrix

Confusion matrix presents the prediction results with a bird's eye view by graphically visualizing the number of examples correctly and incorrectly classified for each class.

4. Classification Report

- Precision: Denotes the number of true positives that were accurately forecasted. (5.3) shows the formula for precision:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{5.3}$$

- Recall (Sensitivity): Refers to the number of true positives predicted correctly. Recall

  is identified as (5.4):

$$\text{Recall} = \frac{TP}{TP + FN} \tag{5.4}$$

- F1-Score: The harmonic mean of precision and recall, both weighted. The below equation (5.5) shows F1 Score:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.5}$$

CHAPTER 6

RESULTS AND DISCUSSION

The Multi-Augmented GoogleNet PCA with DBNs had the highest accuracy of 0.8820 and is therefore the best model. The performance is a demonstration of the strength in using GoogleNet as the feature extractor, PCA for feature dimensionality reduction, and DBNs as the classifier. The performance improvement demonstrates that multi-augmentation helps to enhance feature representation, and PCA maintains important information at lower computational efforts. DBNs are also demonstrated to be stronger when working with extracted features compared to RNNs in this setup. This model has practical applications in real fingerprint recognition systems, such as biometric verification to guarantee secure entry, identification in banks and mobile phones, and forensic applications. With the high accuracy, there is secure and efficient fingerprint classification, thereby reducing misclassification and enhancing security in various applications. Table 6.1 and Table 6.2 demonstrate the multiple models accuracy and loss.

It is analyzed further that the generalization capability of the model is enhanced by taking advantage of the strengths of GoogleNet's feature extraction depth, PCA's reduction in dimensionality, and DBNs' learning of hierarchical representations. The integration not only minimizes computational overhead but also improves resistance to noise and variability in fingerprint images. The approach outperforms other architectures used in testing, thus demonstrating its potential for large-scale biometric systems where security and reliability are of paramount importance. The comparative analysis of various feature extraction and classification methods also reveals the effect of multi-augmentation, showing that diversification of training samples has a significant positive influence on the adaptability of the model to novel fingerprint patterns.

Table 6.1 Results of RNN Model

Dataset Name	Feature Extraction	Dimension Reduction	Accuracy
Inverted	AlexNet	PCA	0.6187
Inverted	AlexNet	LDA	0.0938
Inverted	GoogleNet	PCA	0.7281
Inverted	GoogleNet	LDA	0.0750
Multi-Augmented	AlexNet	PCA	0.6695
Multi-Augmented	AlexNet	LDA	0.1328
Multi-Augmented	GoogleNet	PCA	0.8656
Multi-Augmented	GoogleNet	LDA	0.1445

Table 6.2 Results of DBNs Model

Dataset Name	Feature Extraction	Dimension Reduction	Accuracy
Inverted	AlexNet	PCA	0.6031
Inverted	AlexNet	LDA	0.0688
Inverted	GoogleNet	PCA	0.7688
Inverted	GoogleNet	LDA	0.0500
Multi-Augmented	AlexNet	PCA	0.5828
Multi-Augmented	AlexNet	LDA	0.1258
Multi-Augmented	GoogleNet	PCA	0.8820
Multi-Augmented	GoogleNet	LDA	0.1305

CHAPTER 7

CONCLUSION AND FUTURE WORK

In the current research study, various deep learning models were compared based on fingerprint classification using various methods of feature extraction, dimension reduction, and classifier. Out of all the models, Multi-Augmented GoogleNet PCA with DBNs attained maximum accuracy of 0.8820, proving the effectiveness of GoogleNet, PCA, and DBNs ensemble for fingerprint verification. The results indicate that multi-augmentation enhances feature diversity, PCA performs dimension reduction with no loss of critical information, and DBNs provide robust classification performance. The methodology can enhance real-world biometric identification systems significantly, creating secure and correct identity verification for access control, banking security, and forensic investigation. Additionally, the study discovers that the use of augmentation methods in conjunction with a better deep learning approach significantly optimizes the accuracy of classification without a reduction in computational efficiency. The findings highlight the imperative importance of the selection of appropriate feature extraction, dimensionality reduction, and classification methods in designing an optimally balanced trade-off between computational cost and accuracy. Future studies can investigate real-time implementation, further optimization of computational efficiency, and the expansion of this approach to other biometric tasks, such as face and iris recognition. The findings are the foundation for the design of more secure and robust biometric verification systems, which are more tolerant of real-world problems, such as partial fingerprints, noise corruption, and varying illumination conditions.

APPENDIX – 1

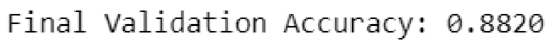
CODING

```
!pip install numpy pillow opencv-python matplotlib tensorflow scikit-learn seaborn
tqdm
import numpy as np
import os
from PIL import Image, ImageOps, ImageFilter
import random
import cv2
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras.applications import VGG16, VGG19, ResNet50, InceptionV3
from tensorflow.keras.layers import Dense, Dropout, GlobalAveragePooling2D
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ModelCheckpoint
from tensorflow.keras.utils import to_categorical
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix
import seaborn as sns
from tensorflow.keras.models import load_model
img_train_path = r'D:\project work 2 phase
2\fingerprint\dataset_FVC2000_DB4_B\dataset\np_data\img_train.npy'
label_train_path = r'D:\project work 2 phase
2\fingerprint\dataset_FVC2000_DB4_B\dataset\np_data\label_train.npy'
train_data_folder = r'D:\project work 2 phase
2\fingerprint\dataset_FVC2000_DB4_B\dataset\train_data'
```

```
def load_npy_file(file_path):
    try:
        data = np.load(file_path)
        print(f"Loaded: {file_path}")
        return data
    except FileNotFoundError:
        print(f"Error: File not found at {file_path}")
        return None
img_train = load_npy_file(img_train_path)
label_train = load_npy_file(label_train_path)
if img_train is not None:
    print(f"Shape of img_train: {img_train.shape}")
if label_train is not None:
    print(f"Shape of label_train: {label_train.shape}")
if img_real is not None:
    print(f"Shape of img_real: {img_real.shape}")
if label_real is not None:
    print(f"Shape of label_real: {label_real.shape}")
def display_images_from_folder(folder_path, num_images=10):
    if not os.path.exists(folder_path):
        print(f"Error: Folder not found at {folder_path}")
    image_files = [f for f in os.listdir(folder_path) if f.lower().endswith(('png', 'jpg',
'jpeg', 'bmp', 'tiff'))]
    if len(image_files) == 0:
        print(f"No image files found in {folder_path}")
        return
    print(f"Displaying up to {num_images} images from {folder_path}...")
    plt.figure(figsize=(10, 5))
    for i, image_file in enumerate(image_files[:num_images]):
        image_path = os.path.join(folder_path, image_file)
        try:
            img = Image.open(image_path)
            plt.subplot(2, 5, i + 1)
            plt.imshow(img)
            plt.axis('off')
            plt.title(f"Image {i + 1}")
        except Exception as e:
            print(f"Error loading image {image_file}: {e}")
    plt.tight_layout()
    plt.show()
display_images_from_folder(train_data_folder, num_images=10)
data_folder = r'D:\project work 2 phase
2\fingerprint\dataset_FVC2000_DB4_B\dataset\train_data'
output_folder = r'D:\project work 2 phase 2\fingerprint\preprocessed_data'
os.makedirs(output_folder, exist_ok=True)
def load_images_with_labels(folder_path):
    if not os.path.exists(folder_path):
        print(f"Error: Folder not found at {folder_path}")
        return pd.DataFrame()
    image_data = []
    labels = []
    image_files = sorted([f for f in os.listdir(folder_path) if f.lower().endswith('.bmp')])
    if len(image_files) == 0:
        print("No .bmp images found in the folder.")
        return pd.DataFrame()
    for img_file in tqdm(image_files, desc="Loading Images"):
        img_path = os.path.join(folder_path, img_file)
        try:
            img = Image.open(img_path).convert('L')
            img_array = np.array(img)
            img_file_clean = img_file.replace('.bmp', '')
            parts = img_file_clean.split('_')
            if len(parts) < 2:  # Ensure both finger_index and user_id exist
                print(f"Skipping invalid file name format: {img_file}")
                continue
            finger_index = int(parts[0])
            user_id = int(parts[1])
            image_data.append(img_array)
            labels.append((finger_index, user_id))
        except Exception as e:
    df = pd.DataFrame({'image': image_data, 'finger_index': [x[0] for x in labels],
'user_id': [x[1] for x in labels]})
    return df
def preprocess_image(image_path):
    img = Image.open(image_path).convert('L')
    img = img.resize((256, 256))
    img_np = np.array(img)
    img_np = cv2.GaussianBlur(img_np, (3, 3), 0)
    img_np = cv2.equalizeHist(img_np)
    return Image.fromarray(img_np)
def preprocess_dataset(folder_path, output_folder):
    if not os.path.exists(folder_path):
        print(f"Error: Folder not found at {folder_path}")
        return
    image_files = [f for f in os.listdir(folder_path) if f.lower().endswith(('.bmp', '.jpg',
'.jpeg', '.png'))]
    if len(image_files) == 0:
        print("No images found in the folder.")
        return
    for img_file in tqdm(image_files, desc="Preprocessing Images"):
        img_path = os.path.join(folder_path, img_file)
        processed_img = preprocess_image(img_path)
        processed_img.save(os.path.join(output_folder, os.path.splitext(img_file)[0] +
"_processed.png"))
image_df = load_images_with_labels(data_folder)
if not image_df.empty:
    print("\nSample Data (Index & User ID):\n")
    print(image_df[['finger_index', 'user_id']].head(10))
    image_df.to_pickle("fingerprint_dataset.pkl")
    print("\nDataset saved as 'fingerprint_dataset.pkl'.")
preprocess_dataset(data_folder, output_folder)
preprocessed_dir = r"D:\project work 2 phase 2\fingerprint\preprocessed_data"
inverted_dir = r"D:\project work 2 phase 2\fingerprint\augmented_data\inverted"
image_df = pd.read_pickle("fingerprint_dataset.pkl")
preprocessed_dir = r"D:\project work 2 phase 2\fingerprint\preprocessed_data"
inverted_dir = r"D:\project work 2 phase 2\fingerprint\augmented_data\inverted"
os.makedirs(inverted_dir, exist_ok=True)
inverted_images = []
inverted_labels = []
def invert_image(image):
    h, w = image.shape  # Get height and width
    half_h = h // 2  # Split height in half
    # Split image into two halves
    top_half = image[:half_h, :]
    bottom_half = image[half_h:, :]
    top_half_flipped = np.flipud(top_half)
    bottom_half_flipped = np.flipud(bottom_half)
    inverted_1 = np.vstack((top_half, top_half_flipped))
    inverted_2 = np.vstack((bottom_half, bottom_half_flipped))
    return inverted_1, inverted_2
for index, row in image_df.iterrows():
    finger_index = f"{row['finger_index']:05d}"
    user_id = f"{row['user_id']:02d}"
    processed_image_filename = f"{finger_index}_{user_id}_processed.png"
    processed_image_path = os.path.join(preprocessed_dir, processed_image_filename)
    if not os.path.exists(processed_image_path):
        print(f"File not found: {processed_image_path}")
        continue
    try:
        original_image = np.array(Image.open(processed_image_path).convert('L'))
    except Exception as e:
        print(f"Error loading image {processed_image_path}: {e}")
```

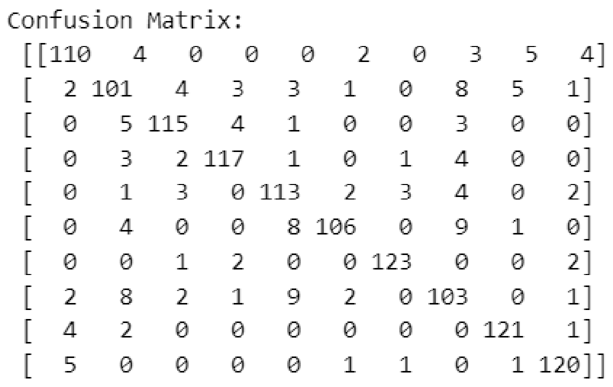
APPENDIX – 2

SCREENSHOTS



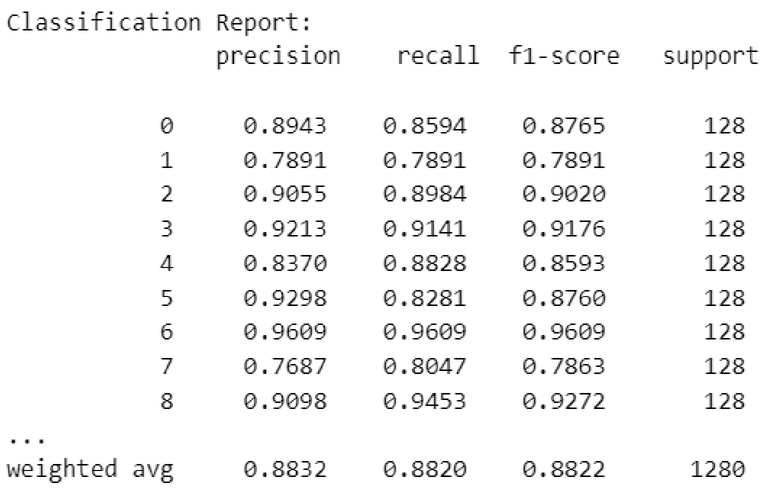
```
Final Validation Accuracy: 0.8820
```

Fig A2.1 Validation Accuracy of Multi-Augmented GoogleNet PCA with DBNs Architecture



```
Confusion Matrix:
 [[110   4   0   0   0   2   0   3   5   4]
 [  2 101   4   3   3   1   0   8   5   1]
 [  0   5 115   4   1   0   0   3   0   0]
 [  0   3   2 117   1   0   1   4   0   0]
 [  0   1   3   0 113   2   3   4   0   2]
 [  0   4   0   0   8 106   0   9   1   0]
 [  0   0   1   2   0   0 123   0   0   2]
 [  2   8   2   1   9   2   0 103   0   1]
 [  4   2   0   0   0   0   0   0 121   1]
 [  5   0   0   0   0   1   1   0   1 120]]
```

Fig A2.2 Confusion Matrix of the Best Model



```
Classification Report:
              precision    recall  f1-score   support

           0     0.8943    0.8594    0.8765       128
           1     0.7891    0.7891    0.7891       128
           2     0.9055    0.8984    0.9020       128
           3     0.9213    0.9141    0.9176       128
           4     0.8370    0.8828    0.8593       128
           5     0.9298    0.8281    0.8760       128
           6     0.9609    0.9609    0.9609       128
           7     0.7687    0.8047    0.7863       128
           8     0.9098    0.9453    0.9272       128
...
weighted avg     0.8832    0.8820    0.8822      1280
```

Fig A2.3 Classification Report of the Best Model

REFERENCES

[1] Chowdhury, A. M., & Imtiaz, M. H. (2022). Contactless fingerprint recognition using deep learning—a systematic review. Journal of Cybersecurity and Privacy, 2(3), 714-730.

[2] Hou, B., Zhang, H., & Yan, R. (2022). Finger-vein biometric recognition: A review. IEEE Transactions on Instrumentation and Measurement, 71, 1-26.

[3] Li, S., Zhang, B., Fei, L., & Zhao, S. (2021). Joint discriminative feature learning for multimodal finger recognition. Pattern Recognition, 111, 107704.

[4] Li, S., Zhang, B., Zhao, S., & Yang, J. (2021). Local discriminant coding based convolutional feature representation for multimodal finger recognition. Information Sciences, 547, 1170-1181.

[5] Militello, C., Rundo, L., Vitabile, S., & Conti, V. (2021). Fingerprint classification based on deep learning approaches: experimental findings and comparisons. Symmetry, 13(5), 750.

[6] Minaee, S., Abdolrashidi, A., Su, H., Bennamoun, M., & Zhang, D. (2023). Biometrics recognition using deep learning: A survey. Artificial Intelligence Review, 56(8), 8647-8695.

[7] Mua’ad, M., Alqadi, Z. A., & Aldebei, K. (2021). Comparative analysis of fingerprint features extraction methods. Journal of Hunan University Natural Sciences, 48(12).

[8] Priesnitz, J., Rathgeb, C., Buchmann, N., Busch, C., & Margraf, M. (2021). An overview of touchless 2D fingerprint recognition. EURASIP Journal on Image and Video Processing, 2021, 1-28.

[9] Rajasekar, V., Predić, B., Saracevic, M., Elhoseny, M., Karabasevic, D., Stanujkic, D., & Jayapaul, P. (2022). Enhanced multimodal biometric recognition approach for smart cities based on an optimized fuzzy genetic

algorithm. Scientific Reports, 12(1), 622.

[10] Ren, H., Sun, L., Guo, J., Han, C., & Wu, F. (2021). Finger vein recognition system with template protection based on convolutional neural network. Knowledge-based systems, 227, 107159.

[11] Shaheed, K., Mao, A., Qureshi, I., Kumar, M., Abbas, Q., Ullah, I., & Zhang, X. (2021). A systematic review on physiological-based biometric recognition systems: current and future trends. Archives of Computational Methods in Engineering, 1-44.

[12] Vijayakumar, T. (2021). Synthesis of palm print in feature fusion techniques for multimodal biometric recognition system online signature. Journal of Innovative Image Processing (JIIP), 3(02), 131-143.

[13] Yang, J., Huang, Y., Zhang, R., Huang, F., Meng, Q., & Feng, S. (2021). Study on ppg biometric recognition based on multifeature extraction and naive bayes classifier. Scientific Programming, 2021(1), 5597624.

[14] Zhang, Z., Wang, Z., Zhang, C., Yao, Z., Zhang, S., Wang, R., ... & Qiu, C. (2024). Advanced terahertz refractive sensing and fingerprint recognition through metasurface-excited surface waves. Advanced Materials, 36(14), 2308453.

[15] Zhang, Z., Zhao, X., Zhang, X., Hou, X., Ma, X., Tang, S., ... & Long, S. (2022). In-sensor reservoir computing system for latent fingerprint recognition with deep.